

This listing of claims will replace all prior versions, and listings, of claims in the application:

Listing of Claims

1. (Currently Amended) A method for communicating with a device, comprising:
executing a kernel module in a memory;
executing at least one kernel thread in the memory to execute device driver functions for the kernel module, wherein the device driver functions are capable of being invoked by system calls from applications operating in a user context; and
executing, with the at least one kernel thread, calls to device driver functions for the kernel module running in a kernel context;
buffering a parameter list;
setting device parameters in the buffered parameter list to values provided by kernel module functions;
setting a flag indicating that the kernel thread needs to set parameters at the device to device parameter values set in the parameter list
processing, by the at least one executed kernel thread, the parameter list by performing:
applying a lock on information in the parameter list including the located buffered parameter values;
after applying the lock, copying the parameter values from the parameter list to a temporary buffer, wherein the device parameters are set to the parameter values from the parameter list in the temporary buffer; and
releasing the lock after copying the parameter values from the parameter list to the temporary buffer.
2. (Original) The method of claim 1, wherein the kernel module spawns at least one kernel thread to execute the calls to the device driver functions for the kernel module.
3. (Original) The method of claim 1, further comprising:
accessing, with one kernel thread, device information from the device; and
buffering the accessed device information.

4. (Original) The method of claim 3, wherein a kernel module function requests device information, further comprising:

in response to the request for the device information, accessing the buffered device information.

5. (Original) The method of claim 1, wherein the kernel thread accesses buffered device information periodically and independently of kernel module requests for the device information.

6. (Canceled)

7. (Canceled)

8. (Currently Amended) The method of claim [[6]] 1, further comprising:
spawning a kernel thread to set device parameters to parameter values buffered in the parameter list.

9. (Currently Amended) The method of claim [[7]] 1, wherein the kernel thread spawned to set device parameter values processes the parameter list to locate buffered parameter values and set the device parameters to the buffered parameter values.

10. (Canceled)

11. (Currently Amended) The method of claim [[10]] 1, further comprising:
disabling higher priority contexts before locking the parameter list; and
enabling the higher priority contexts after releasing the lock on the parameter list.

12. (Original) The method of claim 11, wherein the higher priority context comprises a bottom half or Interrupt Request (IRQ) context.

13. (Currently Amended) The method of claim [[10]] 1, further comprising:

after releasing the lock, executing device driver functions to configure the device with the parameter values in the temporary buffer.

14. (Original) The method of claim 1, further comprising:

initiating, with the kernel module, an access request with respect to device information;
disabling any higher priority contexts capable of accessing the device information;
obtaining a lock for the device information subject to the access request;
providing the kernel module access to the device information;
releasing the lock; and
enabling all higher priority contexts that were disabled.

15. (Currently Amended) A system, comprising:

a network device;

a memory;

a processor executing code to perform:

execute a network device driver in memory to control the network device;

execute a kernel module in the memory;

execute at least one kernel thread in the memory to execute device driver functions for the kernel module, wherein the device driver functions are capable of being invoked by system calls from applications operating in a user context; and

execute, with the at least one kernel thread, calls to device driver functions for the kernel module running in a kernel context;

buffering a parameter list;

setting device parameters in the buffered parameter list to values provided by kernel module functions;

setting a flag indicating that the kernel thread needs to set parameters at the device to device parameter values set in the parameter list;

processing, by the at least one executed kernel thread, the parameter list by performing:

applying a lock on information in the parameter list including the located buffered parameter values;

after applying the lock, copying the parameter values from the parameter list to a temporary buffer, wherein the device parameters are set to the parameter values from the parameter list in the temporary buffer, and
releasing the lock after copying the parameter values from the parameter list to the temporary buffer.

16. (Original) The system of claim 15, wherein the kernel module spawns at least one kernel thread to execute the called device driver functions.

17. (Original) The system of claim 15, wherein the processor executes code to further perform:

access, with one kernel thread, device information from the device; and
buffer the accessed device information.

18. (Original) The system of claim 17, wherein a kernel module function requests device information, wherein the processor executes the code to further perform:
in response to the request for the device information, accessing the buffered device information.

19. (Original) The system of claim 15, wherein the kernel thread accesses device information periodically and independently of kernel module requests for device information.

20. (Canceled)

21. (Canceled)

22. (Currently Amended) The system of claim 15, wherein the kernel thread spawned to set device parameter values processes the parameter list to locate buffered parameter values and set the device parameters to the buffered parameter values.

23. (Canceled)

24. (Currently Amended) The system of claim [[23]] 15, wherein the processor executes the code to further perform:

disabling higher priority context before locking the parameter list; and
enabling the higher priority contexts after releasing the lock on the parameter list.

25. (Currently Amended) The system of claim [[23]] 15, wherein the processor executes the code to further perform:

after releasing the lock, executing device driver functions to configure the device with the parameter values in the temporary buffer.

26. (Original) The system of claim 15, wherein the processor executes the code to further perform:

initiating, with the kernel module, an access request with respect to device information;
disabling any higher priority contexts capable of accessing the device information;
obtaining a lock for the device information subject to the access request;
providing the kernel module access to the device information;
releasing the lock; and
enabling all higher priority contexts that were disabled.

27. (Currently Amended) An article of manufacture for communicating with a device, wherein the article of manufacture causes operations to be performed, the operations comprising:

executing a kernel module;

executing at least one kernel thread to execute device driver functions for the kernel module, wherein the device driver functions are capable of being invoked system calls from applications operating in a user context; [[and]]

executing, with the at least one kernel thread, calls to device driver functions for the kernel module running in a kernel context;

buffering a parameter list;

setting device parameters in the buffered parameter list to values provided by kernel module functions;

setting a flag indicating that the kernel thread needs to set parameters at the device to device parameter values set in the parameter list; and
processing, by the at least one kernel thread, processes the parameter list by performing:
applying a lock on information in the parameter list including the located buffered parameter values;
after applying the lock, copying the parameter values from the parameter list to a temporary buffer, wherein the device parameters are set to the parameter values from the parameter list in the temporary buffer; and
releasing the lock after copying the parameter values from the parameter list to the temporary buffer.

28. (Original) The article of manufacture of claim 27, wherein the kernel module spawns at least one kernel thread to execute the called device driver functions.

29. (Original) The article of manufacture of claim 27, wherein the operations further comprise:

accessing, with one kernel thread, device information from the device; and
buffering the accessed device information.

30. (Original) The article of manufacture of claim 29, wherein a kernel module function requests device information, wherein the operations further comprise:

in response to a request for the device information, accessing the buffered device information.

31. (Original) The article of manufacture of claim 27, wherein the kernel thread accesses buffered device information periodically and independently of kernel module requests for device information.

32. (Canceled)

33. (Canceled)

34. (Original) The article of manufacture of claim 27, wherein the kernel thread spawned to set device parameter values processes the parameter list to locate buffered parameter values and set the device parameters to the buffered parameter values.

35. (Canceled)

36. (Currently Amended) The article of manufacture of claim ~~[[35]]~~ 27, wherein the operations further comprise:

disabling higher priority contexts before locking the parameter list; and
enabling the higher priority contexts after releasing the lock on the parameter list.

37. (Original) The article of manufacture of claim 36, wherein the higher priority context comprises a bottom half or Interrupt Request (IRQ) context.

38. (Currently Amended) The article of manufacture of claim ~~[[35]]~~ 27, wherein the operations further comprise:

after releasing the lock, executing device driver functions to configure the device with the parameter values in the temporary buffer.

39. (Original) The article of manufacture of claim 27, wherein the code executes operations to further perform:

initiating, with the kernel module, an access request with respect to device information;
disabling any higher priority contexts capable of accessing the device information;
obtaining a lock for the device information subject to the access request;
providing the kernel module access to the device information;
releasing the lock; and
enabling all higher priority contexts that were disabled.

REMARKS/ARGUMENTS

Applicants amended the Abstract to overcome the Examiner objection on page 4 of the Final Office Action by adding a period at the end of the sentence.

Applicants amended claims 8, 9, 11, 13, 22, 24, 25, 36, and 38 to depend from one of independent claims 1, 15, and 27 so they would not depend from the canceled claims incorporated into the independent claims.

1. The Amended Claims Comply with 35 U.S.C. §112, par. 1

The Examiner rejected the claims under 35 U.S.C. §112, par. 1 on the grounds that independent claims 1, 15, and 27 recite that “the device driver functions are capable of being invoked system calls from applications operating in a user context” when they should be interpreted to mean that “the device driver functions are capable of being invoked by system calls from applications operating in a user context”. (Final Office Action, pg. 5) The Examiner interpreted this statement to require “invoked by system calls” instead of “invoked system calls”.

In response, Applicants have amended claims 1, 15, and 27 to conform to the Examiner’s interpretation by changing “invoked system calls” to “invoked by system calls”. Applicants request the Examiner to enter this amendment after final because the amendment conforms to the Examiner’s understanding and to expedite prosecution.

2. Independent Claims are Amended to Incorporate Allowable Subject Matter

In the Office Action Summary, the Examiner objected to claims 10, 23, and 35, and in the body of the Office Action found that claim 10 would be allowable if rewritten in independent form. (Final Office Action, pg. 5)

Applicants amended claim 1 to include the requirements of allowable claim 10 and intervening claims 6 and 7 to place claim 1 in condition for allowance. Applicants further amended claim 15 to include the requirements of claim 23 and intervening claims 20 and 21, so as to substantially include in claim 23 the requirements of amended claim 10 and intervening in “system” form. Applicants further amended claim 27 to include the requirements of claim 35 and intervening claims 33 and 32 so that amended claim 27 substantially includes the requirements of amended claims 1 and 23 in “article of manufacture” form.